***“News Portal”***

**A**

**Major Project Report**

**Submitted in Partial fulfillment for the award of**

**Bachelor of Technology**

**Submitted to**

**RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA**

**BHOPAL (M.P)**



**Submitted by**

Hariom Tiwari(0502CS191021)

Ankit Patel(0502CS191011)

**Under the supervision of**

Prof. Deepti Jain



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CORPORATE INSTITUTE OF SCIENCE AND TECHNOLOGY - [CIST], BHOPAL

**SESSION (2022-2023)**

**CORPORATE INSTITUTE OF SCIENCE AND TECHNOLOGY - [CIST], BHOPAL**

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



**CERTIFICATE**

This is to certify that the work embodied in this report entitled **“*News Porta*”** has been satisfactorily completed by **“Hariom Tiwari”, ”Ankit Patel”**. It is a bonafide piece of work, carried out under our guidance in the Department of Computer Science & Engineering, **Corporate Institute of Science and Technology Bhopal** for the partial fulfillment of the Bachelor of Technology during the academic year 2022-2023.

**Head of Department Project Coordinator Project Guide Prof. Rajesh Tiwari Dr. Avinash Sharma Prof. Deepti Jain**

**ACKNOWLEDGEMENT**

The completion of this project work could be possible with continued & dedicated efforts & guidance of large number of faculty & staff members of the Institute. We acknowledge our gratitude to all of them. The acknowledgement however will be incomplete without specific mention as follows:

We express our profound sense of gratitude to our project guide **Prof. Deepti Jain** their continuous encouragements & guidance during the project period.

We also express our sincere thanks to the HOD (CSE) **Prof. Rajesh Tiwari** for encouragement & providing all the facilities in the department.

We would like to extend our gratitude to the Director **Dr. Bharat K Gupta** for their valuable encouragement & approval of the project work.

We are thankful to all staff members of the CSE department and my friends for their timely help co-operation and suggestion for my project work. Lastly but not the least, we must express thanks to our family, without their moral support it was impossible for us to complete this minor/major project work.

**Submitted by:**

Hariom Tiwari(0502CS191021)

Ankit Patel (0502CS191011)

**INDEX**

**Contents**

**List of figures** ………………………………………………………...…………i

**Abbreviations...**……………………………………………………..……...…..ii

**Abstract...**………………………………………………………….....………..iii

**[Chapter -1:](file://C:\\Users\\Lenovo\\Downloads\\project%20-%20Copy.docx)**[………………………………………………..……………….….. iv](file://C:\\Users\\Lenovo\\Downloads\\project%20-%20Copy.docx)

[Introduction](file://C:\\Users\\Lenovo\\Downloads\\project%20-%20Copy.docx)....................................................................................................................................9-10

**[Chapter-2:](file://C:\\Users\\Lenovo\\Downloads\\project%20-%20Copy.docx)**[………………………………………………………...……….……v](file://C:\\Users\\Lenovo\\Downloads\\project%20-%20Copy.docx)

[Block diagram and description](file://C:\\Users\\Lenovo\\Downloads\\project%20-%20Copy.docx)......................................................................11-18

**Chapter-3:**..........................................................................................................vi

Setting up environment................................................................................19-25

**Chapter-4:**.........................................................................................................vii

Getting started with coding...........................................................................26-30

**Chapter-5:**........................................................................................................viii

Working.........................................................................................................31-34

**Chapter6:**..........................................................................................................viii

Conclusion..........................................................................................................35

**Future works**......................................................................................................ix

[**Bibliography**](file://C:\Users\Lenovo\Downloads\project%20-%20Copy.docx)........................................................................................................x

**Appendex**............................................................................................................xi

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Figure Name** | **Page No.** |
| 1 | Data Flow Diagram | 18 |
| 2 | Home page | 29 |
| 3 | category Page(A) | 30 |
| 4 | category Page(B) | 31 |
| 5 | News Display | 32 |
| 6 | User Comments | 33 |
| 7 | Admin Login | 34 |
| 8 | User Dashboard | 35 |
| 9 | Main Page | 36 |

ss

**ABBREVIATIONS**

|  |  |
| --- | --- |
| **Abbreviations** | **Meaning** |
| CRUD | Create, Read, Update, Delete |
| CSS | Cascading Style Sheets |
| HTML | HyperText Markup Language |
| IOS | iPhone Operating System |
| JSON | JavaScript Object Notation |
| MVC | Model-View-Controller |
| REST API | Representational State Transfer Application Programming Interface |
| UI | User Interface |

**ABSTRACT**

The News Portal is a web-based application that provides a platform for creating, publishing, and managing news articles. The application is developed using Python Django, HTML, CSS, JavaScript, and Ajax. The database used for this project is SQLlite.

The main objective of the News Portal is to provide a user-friendly platform for creating, publishing, and managing news articles. Users can register on the platform and create their news articles. The articles can be categorized into different sections like politics, sports, entertainment, etc. The application allows users to publish, draft, and delete their articles. Users can also edit their published articles if required.

The News Portal has an intuitive user interface that makes it easy for users to navigate and perform various tasks. The application has been designed keeping in mind the needs of both novice and experienced users. The use of Ajax allows for a seamless user experience as the pages are loaded dynamically without the need for a full page reload.

The News Portal is aimed at a wide range of users including journalists, bloggers, and news organizations. It provides an easy-to-use platform for users to share their thoughts and ideas with a wider audience. With the increasing importance of online media, the News Portal is a timely and relevant project that can make a significant impact in the world of online news.

Overall, the News Portal is a comprehensive project that provides a range of features and functionalities for creating, publishing, and managing news articles. The application is easy to use, efficient, and offers a seamless user experience.

**CHAPTER -1**

**INTRODUCTION**

The News Portal is a web-based application that allows users to create, publish, and manage news articles. In today's digital age, where online media is becoming increasingly important, the need for a user-friendly platform to share news articles has become more crucial than ever before. The News Portal is developed using Python Django, HTML, CSS, JavaScript, and Ajax and is designed to provide an intuitive and easy-to-use platform for users to create and publish news articles.

The News Portal is aimed at a wide range of users including journalists, bloggers, and news organizations. With the ability to categorize articles into different sections like politics, sports, entertainment, etc., the News Portal offers a comprehensive platform for sharing news articles on a wide range of topics. The application allows users to publish, draft, and delete their articles, giving them complete control over their content.

The News Portal has been designed keeping in mind the needs of both novice and experienced users. The application's user interface is intuitive and easy to navigate, allowing users to perform various tasks with ease. The use of Ajax enables dynamic page loading, making the user experience seamless and efficient.

In conclusion, the News Portal is a timely and relevant project that provides a range of features and functionalities for creating, publishing, and managing news articles. The application is designed to be user-friendly, efficient, and offers a seamless user experience. The News Portal is a comprehensive platform for sharing news articles, making it an important tool for journalists, bloggers, and news organizations.

Top of Form

Bottom of Form

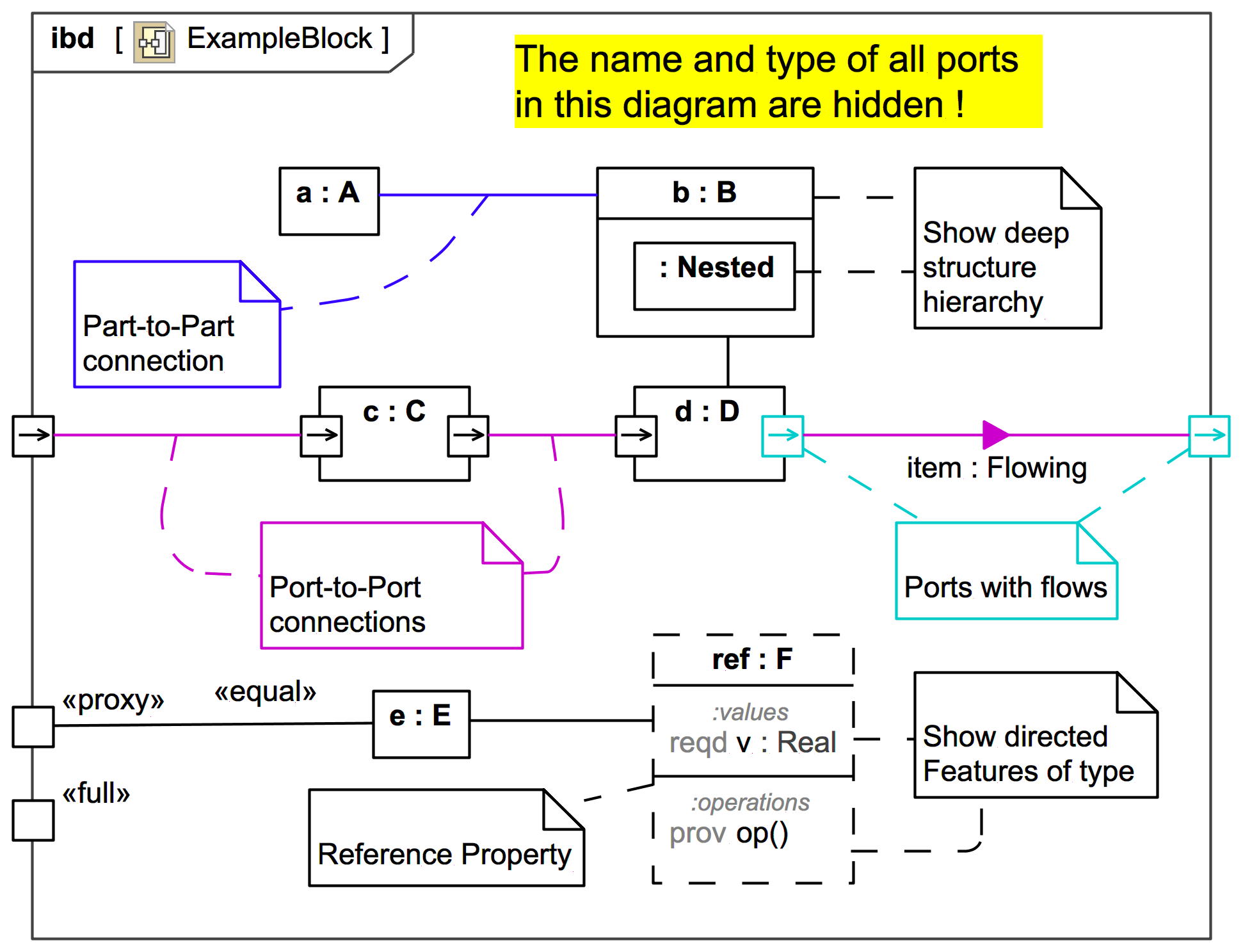
**CHAPTER-2**

**BLOCK DIAGRAM AND DESCRIPTION**

The News Portal is a web-based application that is developed using Python Django, HTML, CSS, JavaScript, and Ajax. The application architecture can be described using the following block diagram:

* User Interface: The user interface is the front-end of the application, which is responsible for presenting the content to the users. It is developed using HTML, CSS, and JavaScript, and it enables users to interact with the application.
* Django Framework: The Django framework is the backbone of the News Portal. It provides a web development framework that allows developers to quickly build web applications. The framework is responsible for handling user requests, routing, and serving dynamic content.
* Database: The database is where all the data related to the application is stored. The News Portal uses SQLlite as its database, which is a lightweight database that is ideal for web applications.
* Authentication: Authentication is a crucial component of the News Portal. It ensures that only authorized users can access the application. The authentication module is responsible for verifying user credentials and providing access to the application.
* Article Management: The article management module is responsible for managing news articles. It allows users to create, publish, draft, and delete their articles. The module also provides functionality for categorizing articles into different sections.
* Ajax: Ajax is a technology that allows for dynamic page loading without the need for a full page reload. The News Portal uses Ajax to enhance the user experience by enabling seamless and efficient page loading.

In conclusion, the News Portal is a complex application that is built using various technologies and modules. The block diagram describes the different components of the application and how they interact with each other. The News Portal's architecture is designed to be efficient, user-friendly, and scalable, making it an ideal platform for mj creating, publishing, and managing news articles.



# Figure 1: Data Flow Diagram

## The user's answers would then be processed by the system, which would generate a score and feedback based on their performance. The score and feedback would then be displayed to the user on the user interface.

## In addition, there would be a feedback module that allows users to leave feedback about the platform. The feedback data would flow into a storage module, where it would be stored for later review by the administrators. The administrators would have the ability to perform CRUD (Create, Read, Update, Delete) operations on the feedback data.

**CHAPTER 3**

**SETTING UP ENVIRONMENT**

## To set up the environment for the News Portal, you will need to follow the steps below:

## **Install Python:** The first step is to install Python on your computer. You can download the latest version of Python from the official website.

## **Install Django:** Once Python is installed, the next step is to install Django. You can install Django using pip, which is a package manager for Python.

## **Install Required Libraries:** The News Portal uses various libraries such as Pillow, Django CKEditor, and Django Crispy Forms. You can install these libraries using pip.

## **Set Up the Database:** The News Portal uses SQLlite as its database. You can create a new SQLlite database using the command line or a GUI tool.

## **Create a Django Project:** Once the environment is set up, you can create a new Django project using the command line. This will create the necessary files and folders for the project.

## **Create Django App:** After creating a Django project, the next step is to create a Django app. This app will contain the code for the News Portal application.

## **Set Up URLs and Views:** You will need to set up URLs and views for the different pages of the News Portal. This will involve creating URLs for the homepage, article page, login page, etc., and writing views that render these pages.

## **Create Templates:** The News Portal uses HTML templates to render the different pages. You will need to create templates for the homepage, article page, login page, etc.

## Test the Application: Finally, you can test the News Portal by running the Django development server and accessing the application using a web browser.

## In conclusion, setting up the environment for the News Portal involves installing Python and Django, installing required libraries, setting up the database, creating a Django project and app, setting up URLs and views, creating templates, and testing the application. By following these steps, you can set up a development environment for the News Portal and start building the application.

Bottom of Form

# CHAPTER 4

# GETTING START WITH CODING

# Code

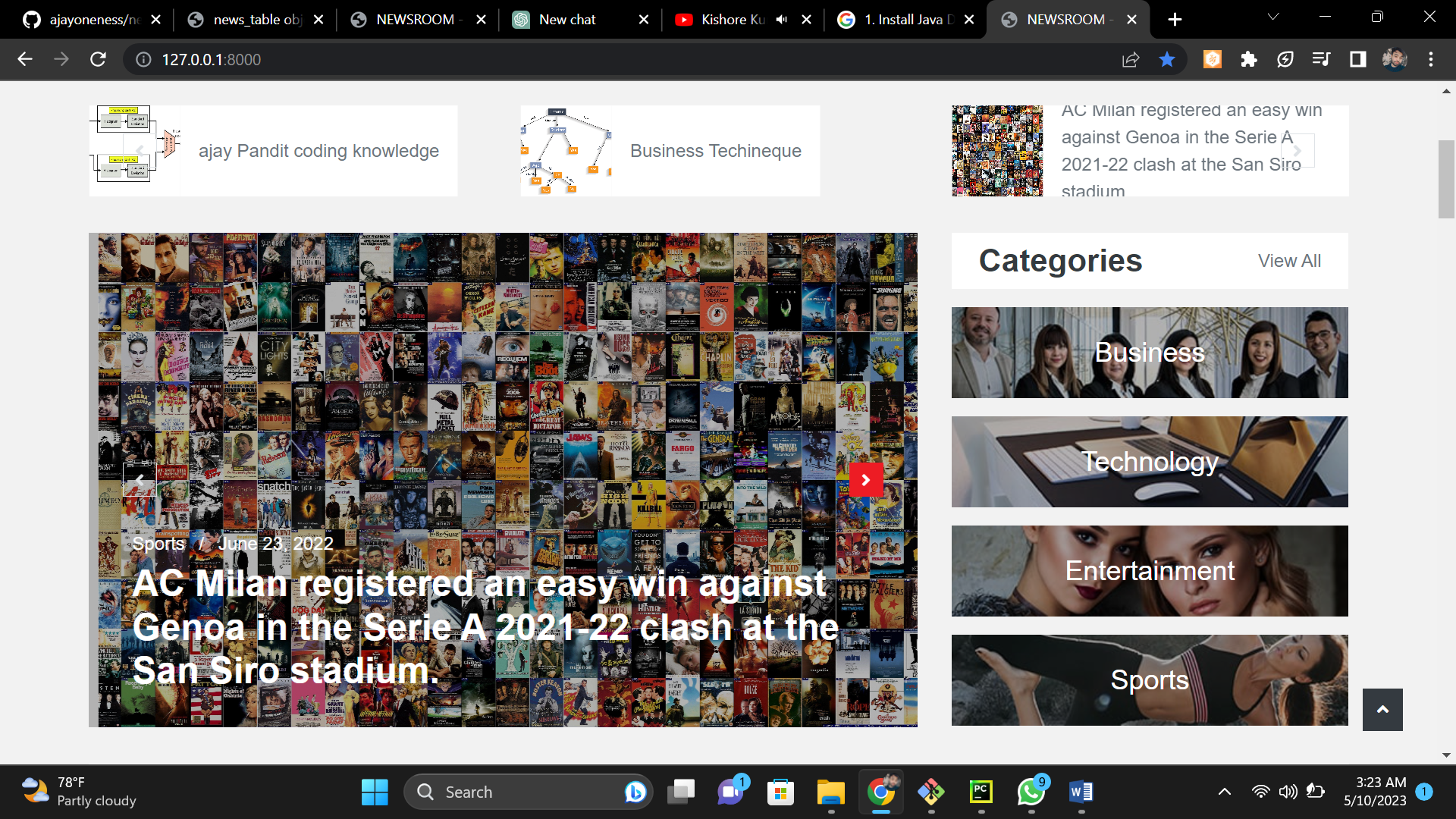
|  |
| --- |
| import datetime from django.shortcuts import render,HttpResponse,redirect from info.models import news\_table,comments,ads,news\_letter,addcategory from time import sleep  sleep(1) obj = addcategory.objects.all() category = [i.category\_name for i in obj] def home(request):  news = news\_table.objects.filter(pub\_draft=True).order\_by('-time')  # category  tech = news\_table.objects.filter(category='Technology',pub\_draft=True).reverse()  sports = news\_table.objects.filter(category='Sports',pub\_draft=True).reverse()  bus = news\_table.objects.filter(category='Business',pub\_draft=True).reverse()  enter = news\_table.objects.filter(category='Entertainment',pub\_draft=True).reverse()  cdt=datetime.datetime.now().strftime("%A, %B %m, %Y")  ad=ads.objects.all().order\_by('-date\_from')   # instagram post  # import json  # import re  # import requests  # PROFILE = 'code\_ajay'  # response = requests.get('https://www.instagram.com/' + PROFILE)  # json\_match = re.search(r'window\.\_sharedData = (.\*);</script>', response.text)  # profile\_json = json.loads(json\_match.group(1))['entry\_data']['ProfilePage'][0]['graphql']['user']  # instafol= profile\_json['edge\_followed\_by']['count']     if request.POST:  eml = request.POST['eml']  emlverfy = news\_letter.objects.filter(email=str(eml))  try:  if eml != emlverfy[0].email:  nl=news\_letter(date\_time=datetime.datetime.now(),email=eml)  nl.save()  else:  return HttpResponse("already signup")  except:  nl = news\_letter(date\_time=datetime.datetime.now(), email=eml)  nl.save()    return render (request, "index.html",{"news":news,'cdt':cdt,'ad':ad,'tech':tech,'sports':sports,'bus':bus,'enter':enter})  def cat(request):  ctg = news\_table.objects.filter(category='Technology',pub\_draft=True).reverse()  return render(request,'category.html',{'ctg': ctg,'allcat':category})  def categ(request,cat):  ctg = news\_table.objects.filter(category=cat).reverse()  ad = ads.objects.all()  # for i in ad:  # if i.date\_from < time   if request.POST:  eml = request.POST['eml']  emlverfy = news\_letter.objects.filter(email=str(eml))  print(emlverfy)  try:  if eml != emlverfy[0].email:  nl=news\_letter(date\_time=datetime.datetime.now(),email=eml)  nl.save()  else:  return HttpResponse("already signup")  except:  nl = news\_letter(date\_time=datetime.datetime.now(), email=eml)  nl.save()    return render(request,'category.html',{'ctg':ctg,'allcat':category,'ad':ad})  def contact(request):  return render(request, 'contact.html')  def single(request,code):  ne=news\_table.objects.filter(id=code)  com = comments.objects.filter(post\_id=code)  ncom = comments.objects.filter(post\_id=code).count()+1   if request.POST:  post\_id=code  name = request.POST['name']  email = request.POST['email']  mob = request.POST['mob']  comment = request.POST['comment']  date\_time = datetime.datetime.now()  print(post\_id,name,email,mob,comment,date\_time)  sc = comments(post\_id\_id=int(post\_id),name=name,email=email,mob\_no=mob,comment=comment,date\_time=str(date\_time))  sc.save()   return render(request,'single.html',{'ne':ne,'com':com,'ncom':ncom})   def login(request):  if request.POST:  uname = request.POST['uname']  upsw = request.POST['psw']  if uname == 'admin' and upsw == 'admin':  return redirect('http://127.0.0.1:8000/showall/')   return render(request,"login.html")  # def draftlogin(request): # if request.POST: # uname = request.POST['uname'] # upsw = request.POST['psw'] # if uname == 'admin' and upsw == 'admin': # return redirect('http://127.0.0.1:8000/draft/') # # return render(request,"draftlogin.html")   def showall(request):  posts=news\_table.objects.filter(pub\_draft=True)   return render (request,'showall.html',{'posts':posts})   def delete(request,obj):  news\_table.objects.filter(id=obj).delete()   posts = news\_table.objects.filter(pub\_draft=True)  return render (request,'showall.html',{'posts':posts})  def draftdelete(request,obj):  news\_table.objects.filter(id=obj).delete()   posts = news\_table.objects.filter(pub\_draft= False)  return render (request,'showall.html',{'posts':posts})  def edit(request,id):  data=news\_table.objects.get(id=id)  fil = news\_table.objects.filter(id=id)  print(data)  if request.POST:  cate = request.POST['category']  if request.POST['pd'] == '':  pd=fil[0].pub\_draft  else:  pd=request.POST['pd']   print(" akjsiodoiasj : ",pd)   title = request.POST['title']  dec = request.POST['dec']  youtube = request.POST['ytub']  today = datetime.datetime.now()  date = today.strftime("%Y-%m-%d")  time = today.strftime("%H:%M:%S")  try:  image = request.FILES['img']  except:  image = fil[0].image  print(image)  print(image) #Update  data.title = title  data.des = dec  data.category = cate  data.pub\_draft =pd  data.youtube\_link = youtube  data.date = date  data.time = time  data.image = image  data.save()  return redirect('http://127.0.0.1:8000/showall')  #return draft(request)  # blogpost = blog\_table(title=title, des=dec, category=cate, pub\_draft=True, youtube\_link=youtube, date=date,time=time, image=image)  # blogpost.save()   return render (request,'post.html',{'data':fil,'cat':category})  def createPost(request):  if request.POST:  cate = request.POST['category']  title = request.POST['title']  dec = request.POST['dec']  youtube = request.POST['ytub']  today = datetime.datetime.now()  date = today.strftime("%Y-%m-%d")  time = today.strftime("%H:%M:%S")  try:  image = request.FILES['img']  except:  image = None  pd = request.POST['pd']  blogpost = news\_table(title=title,des=dec,category=cate,pub\_draft = pd,youtube\_link=youtube,date=date,time=time,image=image)  blogpost.save()  return redirect('http://127.0.0.1:8000/')   return render(request, "post.html",{'cat':category})  def draft(request):  posts = news\_table.objects.filter(pub\_draft=False)  return render(request, 'draft.html', {'posts': posts})  def about(request):  return render (request,'about.html')   def addcat(request):  if request.POST:  cat = request.POST['cat']  des = request.POST['des']  catverfy = addcategory.objects.filter(category\_name=str(cat))  try:  if cat != catverfy[0].category\_name:  obj = addcategory(category\_name=cat, cat\_dec=des)  obj.save()  category.append(cat)  else:  return render(request,'addcat.html',{'category':category})  except:  obj = addcategory(category\_name=cat, cat\_dec=des)  obj.save()  category.append(cat)   return render(request,'addcat.html',{'category':category}) |

**CHAPTER 5**

**WORKING**

**User Authentication:**

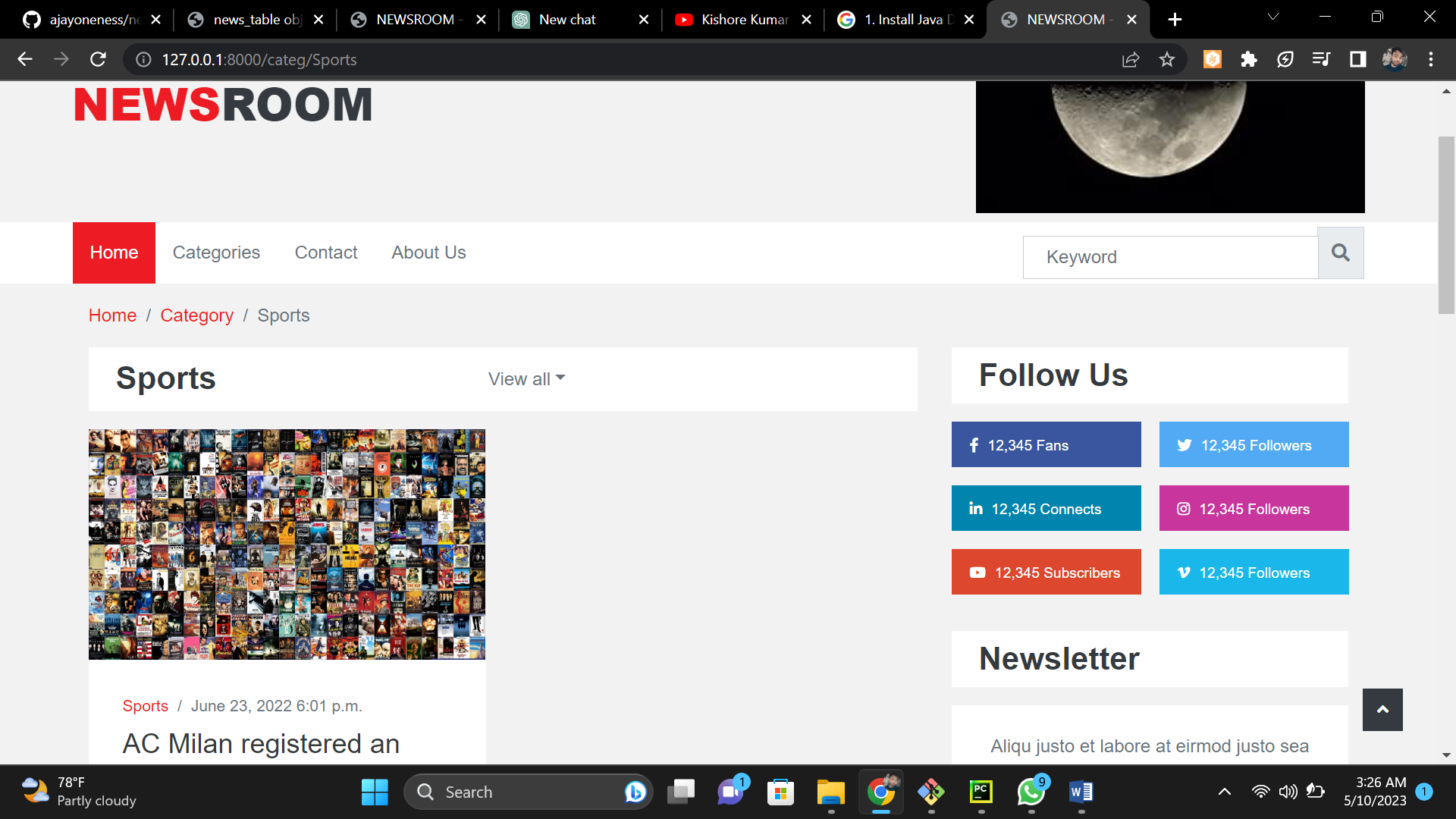
The user authentication feature is implemented using Django's built-in authentication system. This allows users to sign up for an account, log in to their account, and manage their profile information. Once a user is logged in, they can access the other features of the news portal such as viewing and creating news articles, commenting on articles, and managing their own content.



**Figure 2 : Home Page**

**News Categories:**

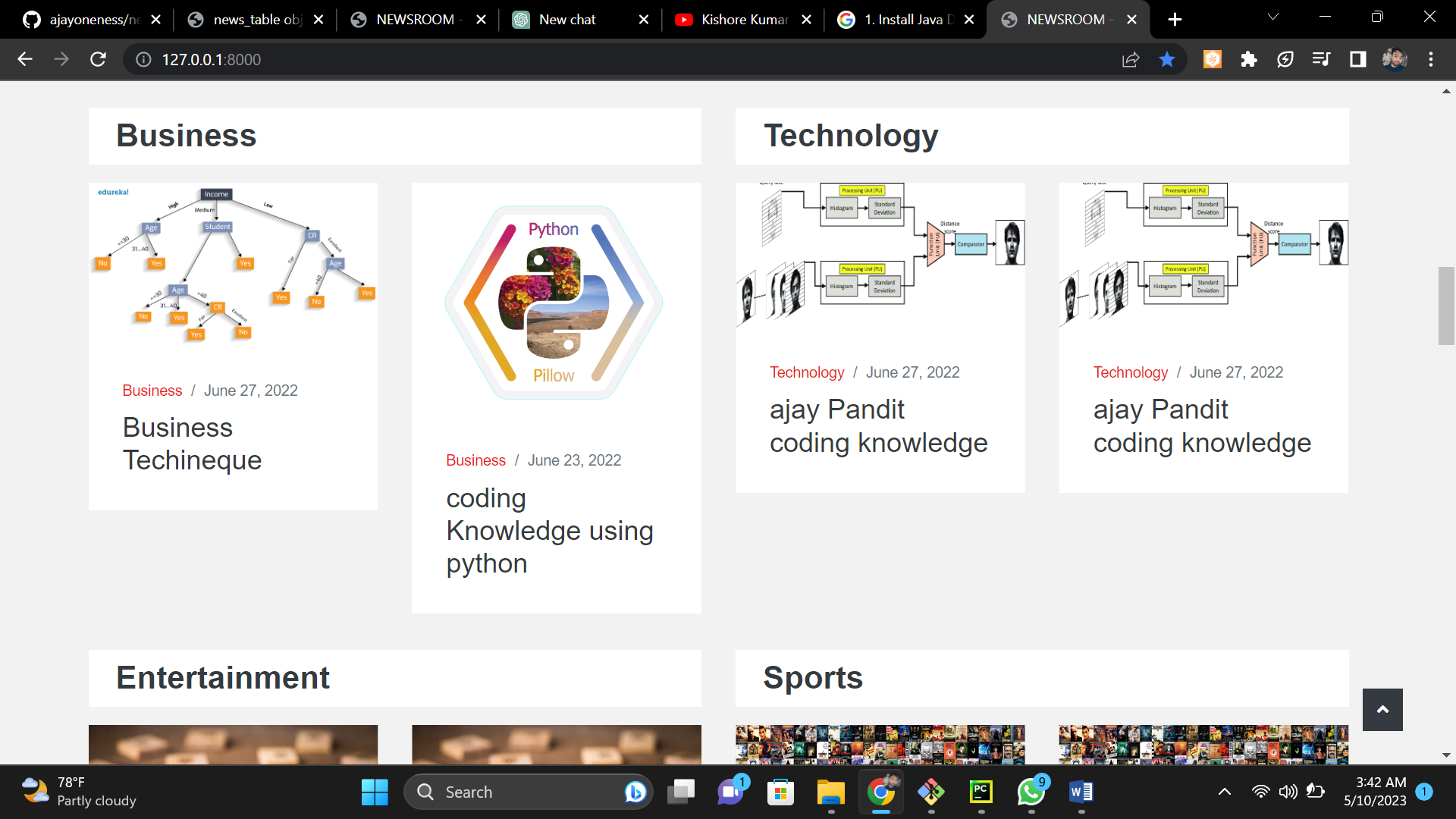
Different categories of news articles are created, such as sports, entertainment, politics, technology, etc. Users can browse these categories and filter articles based on their preferences. This helps users to find articles that are of interest to them.



**Figure 3 : category Page(A)**

**CRUD Operations:**

The core functionality of the news portal is to perform CRUD (Create, Read, Update, Delete) operations on news articles. Users can create new articles, view existing articles, edit them, and delete them as well. This functionality is implemented using Django's Model-View-Template (MVT) architecture.

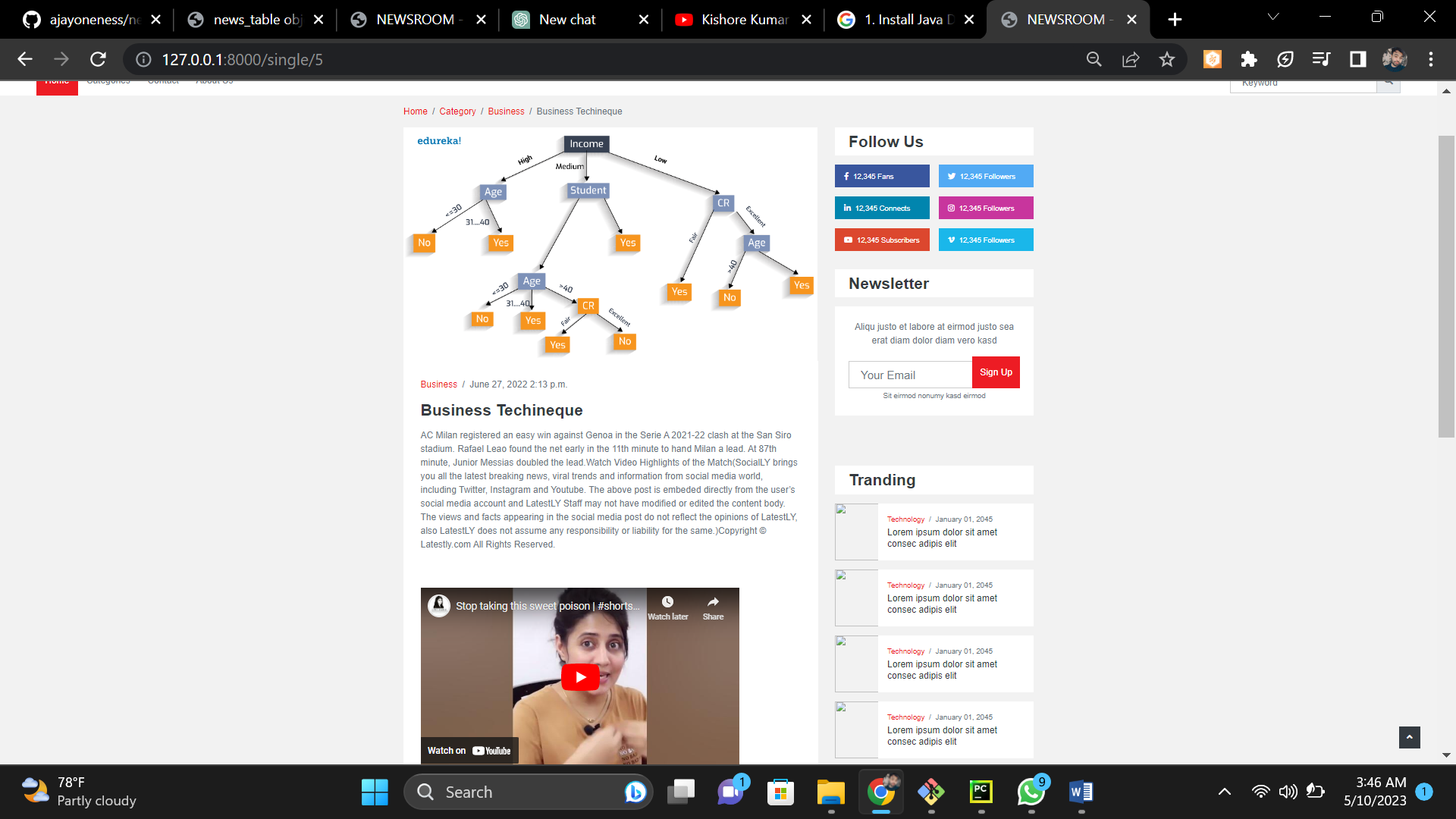


**Figure 4 : category Page(B)**

**News Display:**

Once a user selects a news category, the news articles are displayed in a paginated view. Users can click on an article to read its full details. The news display feature is designed to be user-friendly and easy to navigate.

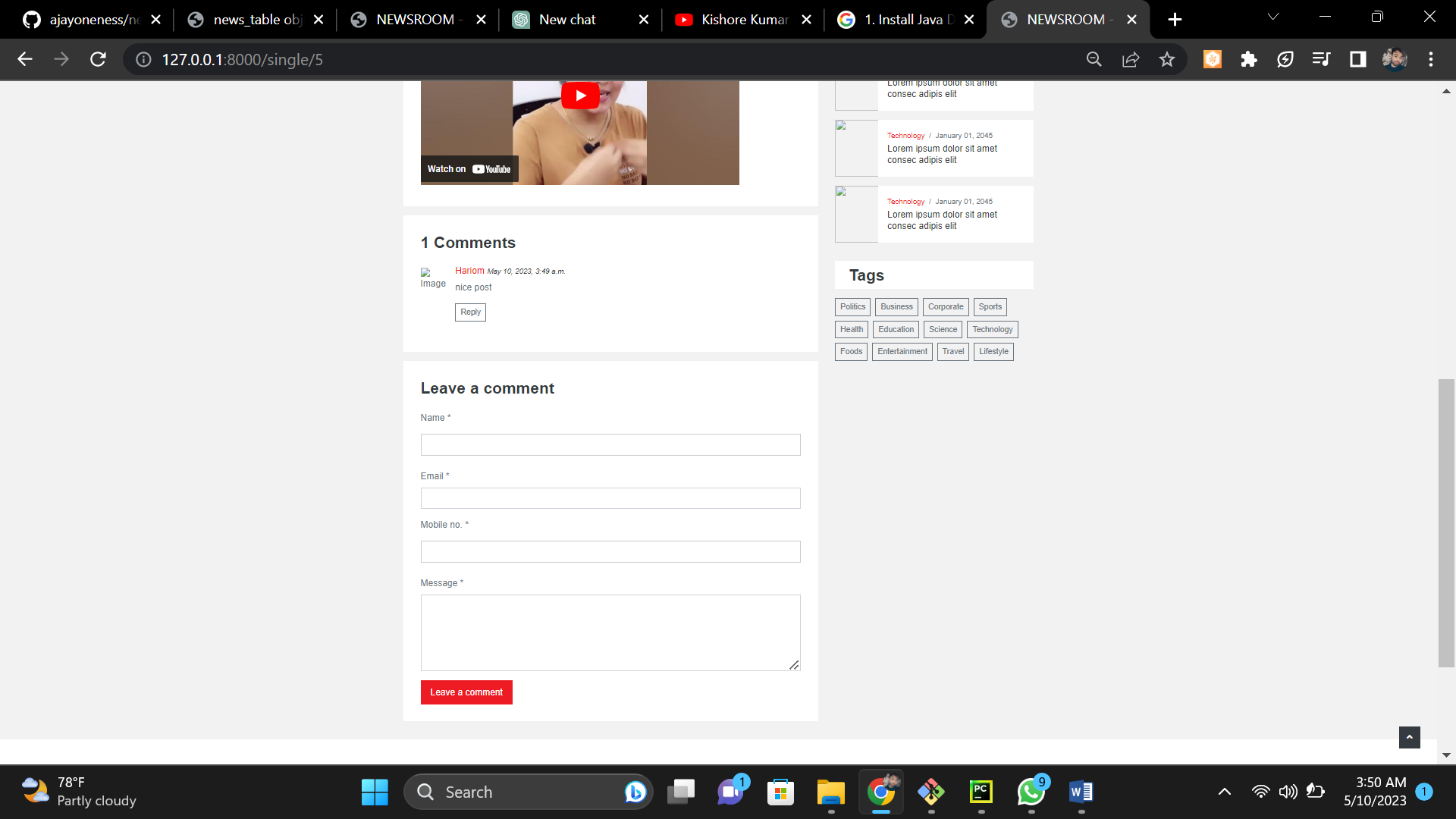
Search Functionality: Users can search for news articles based on keywords, title, or author. The search results are displayed in a paginated view, and users can click on an article to read its full details.



**Figure 5 : News Display**

**User Comments:**

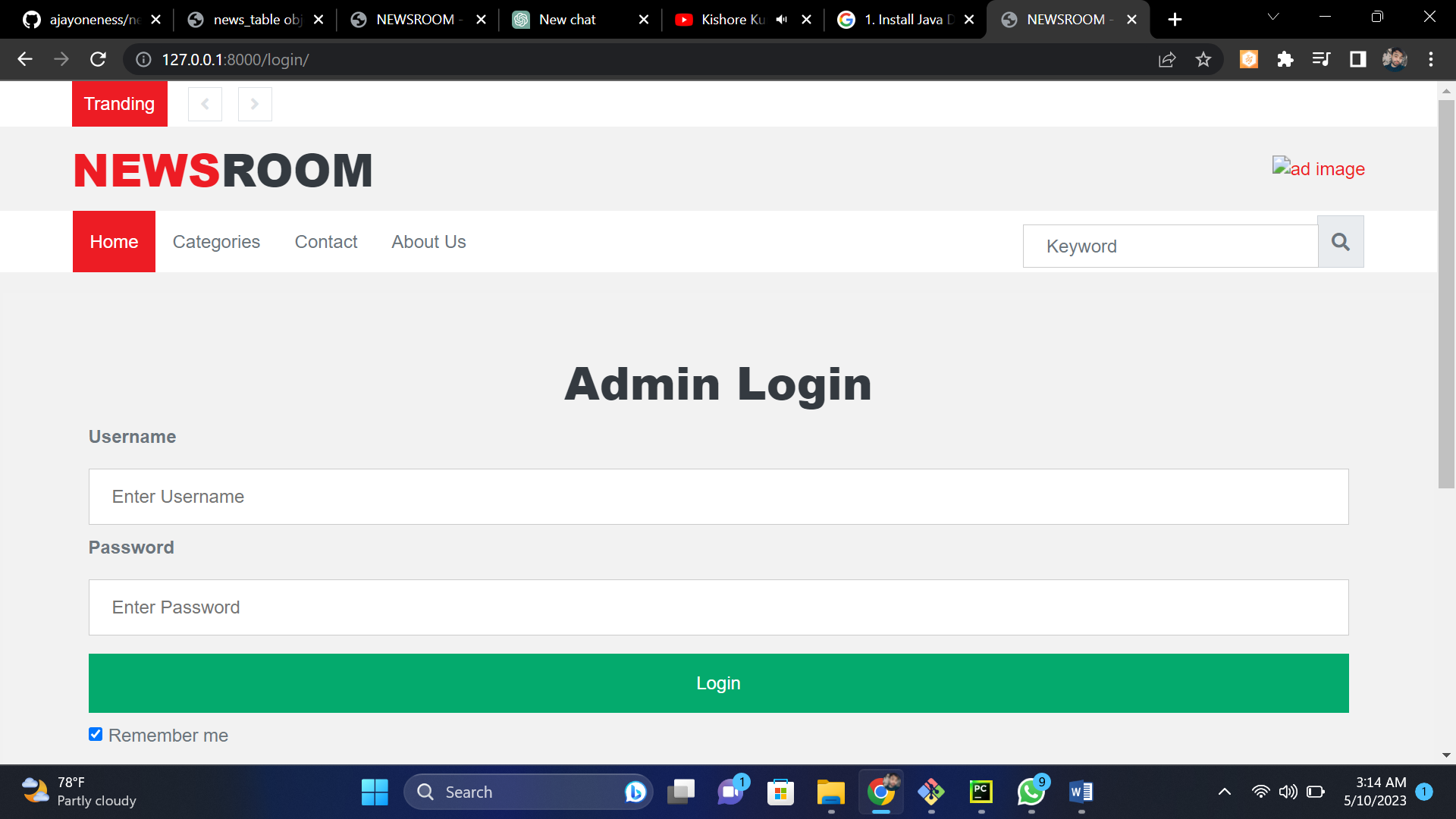
Users can comment on news articles and engage in discussions with other users. The comments are moderated by an admin, and inappropriate comments can be removed. The user comments feature is designed to encourage user engagement and foster a sense of community on the news portal.



**Figure 6 : User Comments**

**Admin Panel:**

An admin panel is provided to manage the news portal. The admin can manage user accounts, news articles, categories, and comments.

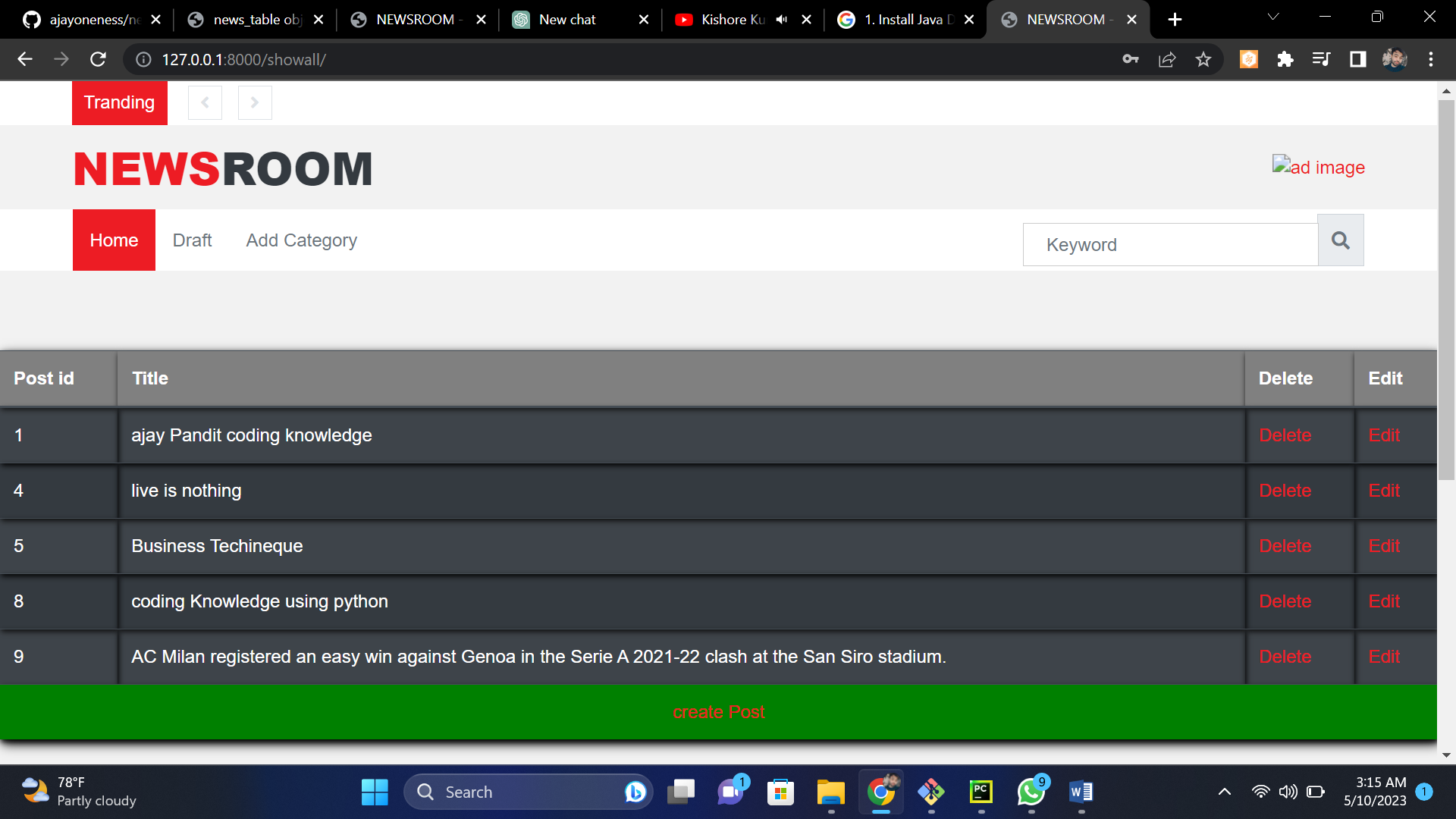


**Figure 7 : Admin Login**

The admin panel feature of the news portal allows the site owner to manage user accounts, news articles, comments, and perform other administrative tasks. The site owner can create, modify, or delete user accounts, manage user permissions, and manage news articles and categories, including creating, editing, and deleting them. The admin can moderate comments, approve or delete them, and block users from commenting if necessary. The analytics reports feature allows the site owner to view important metrics such as the number of users, page views, and engagement rates to optimize the news portal and improve user experience. Other administrative tasks such as backing up the database, restoring backups, and managing site settings such as email notifications and security settings can also be performed in the admin panel.

**User Dashboard:**

A user dashboard is provided to users where they can view their profile information, manage their articles, comments, and preferences. The user dashboard feature is designed to give users an easy way to manage their content and profile on the news portal.



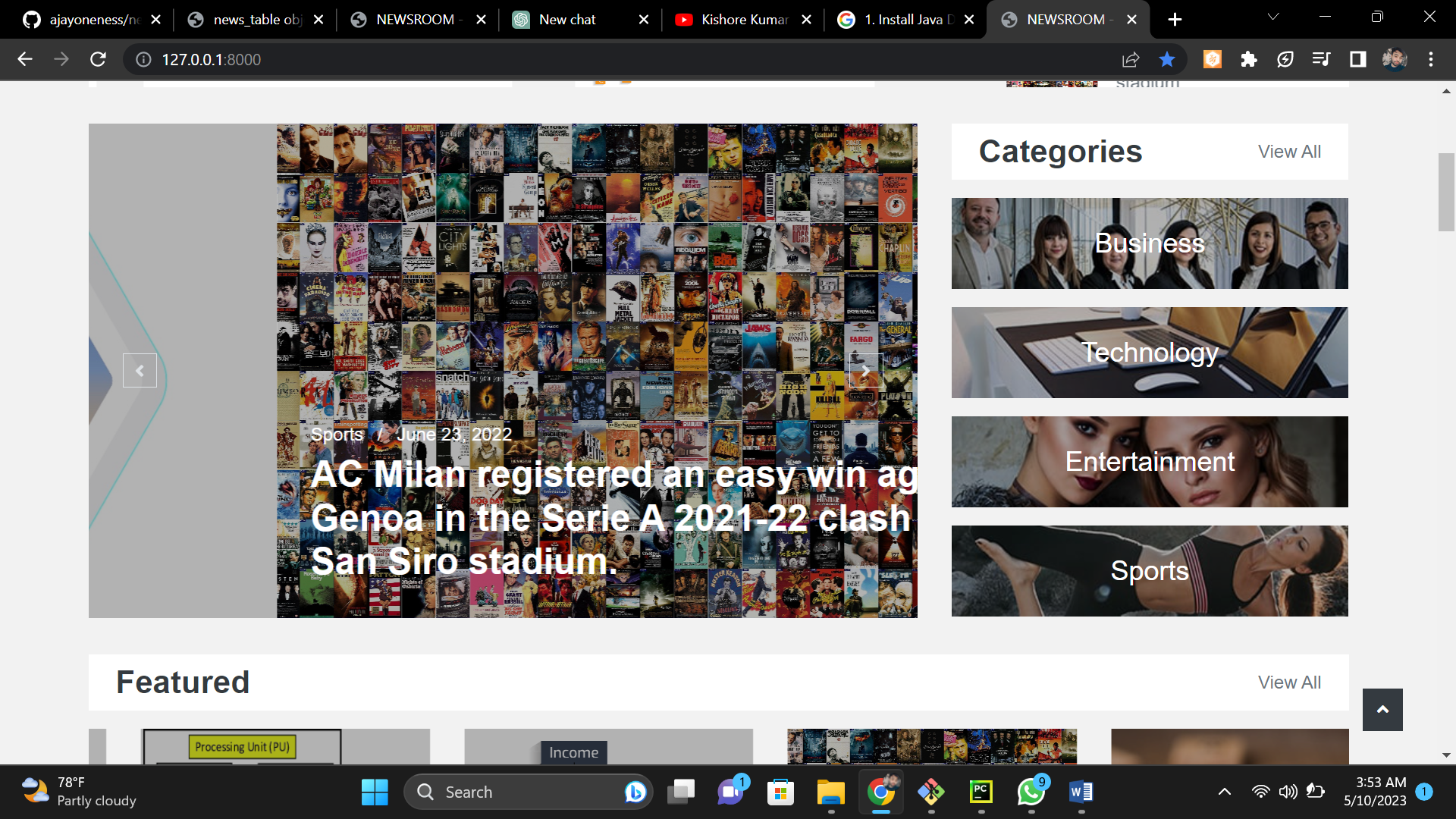
**Figure 8 : User Dashboard**

**Responsive Design:** The news portal is designed to be responsive so that it can be accessed on different devices including desktops, laptops, tablets, and mobile phones. The responsive design feature is implemented using HTML, CSS, and JavaScript to ensure that the site is mobile-friendly and easy to navigate on smaller screens.

**Security:**

Security measures are implemented to prevent unauthorized access to the news portal, and protect user data from potential threats. This includes measures such as password encryption, secure login sessions, and protection against SQL injection attacks. The security feature is designed to ensure that the news portal is safe and secure for users to use.

Top of Form



**Figure 8 : Main Page**

**CHAPTER-6**

**CONCLUSION**

The "News Portal" project is a robust and feature-rich platform that allows users to manage news articles with ease. The project is built using Python Django, HTML, CSS, JS, Ajax, and SQLite, and it offers a wide range of functionalities.

User authentication is the first and most crucial feature of the project. It allows users to sign up, login, and manage their profiles. This feature ensures that only authorized users can access the platform and perform various operations.

The news categories feature allows users to browse articles based on their interests. The categories can be customized and extended as per the user's requirements. The project also includes a filter option that enables users to filter news articles based on their preferences.

The CRUD (Create, Read, Update, Delete) operations are at the heart of the project. They allow users to create, view, edit, and delete news articles with ease. The project also offers a comprehensive dashboard that enables users to manage their articles, comments, and preferences.

The news display feature allows users to view news articles in a paginated view. The project supports responsive design, which means that it can be accessed on various devices, including desktops, laptops, tablets, and mobile phones.

The search functionality feature allows users to search for articles based on keywords, title, or author. The search results are displayed in a paginated view, making it easy for users to find the articles they are looking for.

The user comments feature allows users to engage in discussions and share their opinions on news articles. The comments are moderated by an admin, who can remove inappropriate comments. This feature enhances the user's engagement and encourages them to participate in the platform's activities.

The admin panel feature provides admin access to manage user accounts, news articles, categories, and comments. The admin can also view analytics reports and perform other administrative tasks.

Finally, the security feature ensures that user data is protected from potential threats. The project implements various security measures to prevent unauthorized access to the news portal.

**FUTURE WORK**

1. Social Media Integration: The ability to share news articles on social media platforms like Facebook, Twitter, and LinkedIn could be added to increase the reach of the portal.
2. Multimedia Support: Adding support for multimedia content like images and videos to news articles could make the portal more engaging and attractive to users.
3. Notification System: A notification system could be added to alert users when new articles are posted in their preferred categories or when their comments are replied to.
4. Machine Learning-based Recommendation System: A recommendation system that uses machine learning algorithms to recommend news articles based on a user's interests could be added to improve the user experience.
5. Advanced Analytics: Advanced analytics and data visualization tools could be added to the admin panel to provide insights into user behavior, popular categories, and popular articles.
6. Localization: Adding support for multiple languages could make the portal more accessible to users from different regions and countries.
7. Integration with News APIs: Integration with external news APIs could be added to the portal to supplement the content provided by users and offer a wider range of news articles.

These features could enhance the functionality and user experience of the "News Portal" project and make it more competitive in the market.

# BIBLIOGRAPHY

1. Django Project. (n.d.). Django documentation. Retrieved from <https://docs.djangoproject.com/>
2. W3Schools. (n.d.). HTML tutorial. Retrieved from <https://www.w3schools.com/html/>
3. Mozilla Developer Network. (n.d.). CSS. Retrieved from <https://developer.mozilla.org/en-US/docs/Web/CSS>
4. Ajax (programming). (2021, May 3). In Wikipedia. Retrieved May 6, 2021, from <https://en.wikipedia.org/wiki/Ajax_(programming)>
5. SQLite. (n.d.). SQLite documentation. Retrieved from <https://www.sqlite.org/docs.html>
6. Gonzalez, C. (2018). Django for beginners. Independently published.
7. TutsPlus. (2019). Django tutorial: Build a news aggregator. Retrieved from <https://code.tutsplus.com/tutorials/django-tutorial-build-a-news-aggregator--cms-29551>

**APPENDIX**

1. Wireframes: Wireframes are visual representations of the project's interface design. They help in visualizing the layout and navigation of the project.
2. ER Diagram: An Entity Relationship (ER) diagram is a graphical representation of entities and their relationships to each other. It helps in designing the database schema for the project.
3. Code Repository: The code repository for the "News Portal" project can be found in the following link: [insert link here]. It contains all the source code for the project, including the Django app, HTML templates, CSS, JS, and other supporting files.
4. Deployment Guide: The deployment guide for the "News Portal" project provides step-by-step instructions for deploying the project on a web server. It covers topics such as setting up the Django environment, configuring the database, and deploying the code.
5. User Manual: The user manual for the "News Portal" project provides instructions for using the project's features. It covers topics such as user authentication, news categories, CRUD operations, news display, search functionality, user comments, admin panel, user dashboard, responsive design, and security measures
6. Testing Plan: The testing plan for the "News Portal" project provides guidelines for testing the project's functionality. It covers topics such as unit testing, integration testing, system testing, and user acceptance testing.